March 2022

5G DEPLOYMENT WITH DISTRIBUTED APPROACH FOR NF INSTANCE AND SERVICE DISCOVERY

Niranjan M. M

Nagaraj Kenchaiah

5G DEPLOYMENT WITH DISTRIBUTED APPROACH FOR NF INSTANCE AND SERVICE DISCOVERY

AUTHORS:
Niranjan M M
Nagaraj Kenchaiah

ABSTRACT

In 3rd Generation Partnership Project (3GPP) fifth-generation (5G) network deployments, Network Function (NF) instance and service discovery is done using a centralized method comprising one or more NF Repository Functions (NRFs) which maintain information about all of the registered NF instances and their services. If a NRF fails, then all of the NF instances, and all of the services that are registered with that NRF, will be lost. Such a loss may result in call failures until the failure is detected and recovered using methods such as, for example, a heartbeat (i.e., a keep-alive) mechanism, etc. To address the type of challenge that was described above, techniques are presented herein that employ a private and permissive blockchain-based distributed ledger (which for simplicity of exposition may be referred to herein as a Ledger) to discover NF instances and their services in 5G network deployments. Aspects of the presented techniques decentralize the NF instance and service discovery process without compromising authentication or security. According to aspects of the presented techniques, when a NF instance comes up it may authenticate itself with a blockchain provider to confirm that it is a legitimate NF instance, after which the NF instance and its services may be added to the Ledger. Further, a Ledger is made available to all of the NF instances so that a NF instance may learn about the other NF instances and their services.

DETAILED DESCRIPTION

As an initial matter, it will be helpful to confirm the meaning of a number of the terms that appear in the narrative that is presented below. Specifically:

Term	Meaning
AIK	Attestation Identity Key
BP	Blockchain Provider

Published by Technical Disclosure Commons, 2022

HPLMN	Home PLMN
NF	Network Function
NRF	NF Repository Function
PBFT	Practical Byzantine Fault Tolerance
PCR	Platform Configuration Registers
PK	Public Key
PLMN	Public Land Mobile Network
PoET	Proof of Elapsed Time
SK	Secret Key or Private Key
TPM	Trusted Platform Module (or virtual TPM (vTPM) for a virtual or containerized form)
VPLMN	Visited PLMN

Additionally, it will be helpful to indicate that an annex (Appendix A) may be found at the end of this document. That annex provides additional explanatory material for several of the topics that are discussed in the narrative that is presented below.

In a 3rd Generation Partnership Project (3GPP) fifth-generation (5G) core (5GC) network, a NRF supports a number of functions, including maintaining a NF profile for the available NF instances and their supported services; allowing other NF instances to subscribe to, and be notified regarding, the registration in the NRF of new NF instances of a given type; supporting a service discovery function whereby the NRF receives NF discovery requests from NF instances and provides information on the available NF instances that fulfill certain criteria (e.g., that support a given service).

Figure 1, below, illustrates elements of an exemplary NRF arrangement.

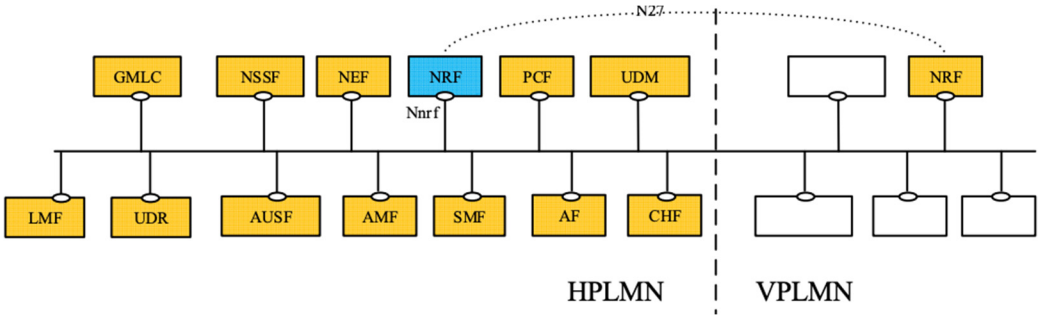

Figure 1: Exemplary NRF Arrangement

To expand upon the NRF description that was presented above, a NRF provides a range of services. First, a NRF allows a NF instance in the serving PLMN to register, update, or deregister its profile in the NRF using representational state transfer (REST) application programming interfaces (APIs). A NF profile consists of the general parameters of a NF instance and also the parameters of the different services that are exposed by the NF instance.

Further, it allows one NRF instance to register, update, or deregister its profile in another NRF in the same PLMN, it allows a NF to subscribe to be notified of newly registered NF instances along with their NF services, and it supports the retrieval of a list of NF instances that are currently registered in the NRF or the NF profile of a given NF instance.

Still further, a NRF allows a NF instance to discover the services that are offered by other NF instances by querying the local NRF, which could be within the same PLMN or across a different PLMN (e.g., a home or a visited PLMN). Additionally, a NRF that is configured with multiple PLMN IDs (i.e., a Mobile Country Code (MCC) and Mobile Network Code (MNC)) supports the registering, updating, and deregistering of the profile of the NF instances from any of those PLMN IDs. Further, multiple NRFs may be deployed in one PLMN where one NRF may query the NF instances in a different NRF so as to fulfil the service discovery request from an NF service consumer. If a NRF receiving a service discovery request does not have the information to fulfil the request, then the query between these two NRFs may be redirected or forwarded by a third NRF.

In summary, a NRF maintains a list of NF instances and the corresponding services that they provide. Whenever it is required, NF instances may discover the services that are offered by other NF instances by querying the local NRF. Additionally, multiple NRFs may be deployed in one PLMN and a NRF may be configured to work with multiple PLMN IDs.

In other words, a NRF is a centralized Service Registry which maintains all of the registered NF service instances and aids in a service discovery process by other NF instances. If a NRF fails, then all of the NF instances, and all of the services that are registered with that NRF, will be lost. Such a loss may result in call failures until it the

failure is detected and recovered using methods such as, for example, a heartbeat (i.e., a keep-alive) mechanism, etc.

In general, there are two main service discovery patterns – a client-side discovery and a server-side discovery. (For additional information regarding these patterns please see the annex (i.e., Appendix A) that may be found at the end of this document). Both of these discovery methods require a centralized Service Registry, which maintains all of the available service instances. Additionally, centrally managed methods are prone to a single point of failure.

Considering the disadvantages of centrally managed methods, a way is needed to decentralize the NF instance and service discovery process in support of identifying all of the other NF instances and their services in order to communicate with them. Further, an authenticated and secure way is needed for communicating such information across all of the NF instances in 5G network deployments.

As described previously, a NRF is a centralized service repository comprising all of the NF instances that are registered with the NRF. There may be multiple NRFs in the same PLMN, and the same NRF may be part of multiple PLMNs.

Currently there are some solutions that handle the issue of NRF scaling using a hierarchical approach and there are also some solutions that handle NRF failure scenarios. However, none of those solutions provide any method for decentralizing a NF instance and service registration process and a subsequent discovery process.

Considering the disadvantages of the centralized methods, a method is needed to decentralize the NF instance and service discovery process in large 5G deployments without compromising authentication and security. As such deployments are on public, private, and hybrid cloud (in a virtual or container form) environments, it is extremely important to ensure that the participating NF instances are authentic and that communication is secured.

In order to address the types of challenges that were described above, techniques are presented herein that provide for the authenticated and secure discovery of NF instances, and of a NF instance's services, in 5G network deployments using a private and permissive blockchain-based distributed ledger (which for simplicity of exposition may be referred to herein as a Ledger). Aspects of the presented techniques address the issue of man-in-the-

middle (MITM) attacks (from, for example, network service spoofing, etc.), aid in moving a service from one NRF or PLMN to another, and also aid in a device or instance replacement in an authenticated and secure manner.

By employing a Ledger, according to aspects of the techniques presented herein, it is possible to ensure that only authenticated NF instances have access to the permissioned Ledger. Authentication using secure credentials would not be sufficient (in the case of trusted deployments) where verification of the trustworthiness and the integrity of the NF instances is necessary before adding to the Ledger.

Regarding trustworthiness, aspects of the presented techniques employ a simple attestation protocol. First, to attest a NF instance, a Blockchain Provider (BP) retrieves the public Attestation Identity Key (AIK) – i.e., PK_AIK_NODE – and it sends a "nonce" to the target NF instance. Second, the target NF instance answers with a quote that contains the "nonce" and its current PCR values that are stored inside of a local TPM (or a vTPM in the case of a virtual or containerized form) of the target NF instance. Third, the quote is signed by the private AIK – i.e., SK_AIK_NODE – of the target NF instance to ensure the integrity of the response. Fourth, following receipt of the payload from the target NF instance, the BP compares the "nonce" to check the freshness of the attestation and determine if this matches the expected value (by comparing the received PCR values with a library of trusted node configurations). Note that the TPM functionality and the library of trusted node configurations may not have hardware details (e.g., in the cases where a NF instance is deployed on a virtual machine or in the cloud).

The techniques presented herein, as described in the above narrative, offer a number of points of interest, including, among other things, authentication of NF instances providing the services using a Ledger; secure NF instance and service registration and a service transfer from one NRF or PLMN to another in 5G network deployments; and an efficient, scalable, and immutable distributed method for discovering the services in 5G deployments.

Figure 2, below, depicts elements of an exemplary architecture according to aspects of the techniques presented herein and reflective of the above narrative.

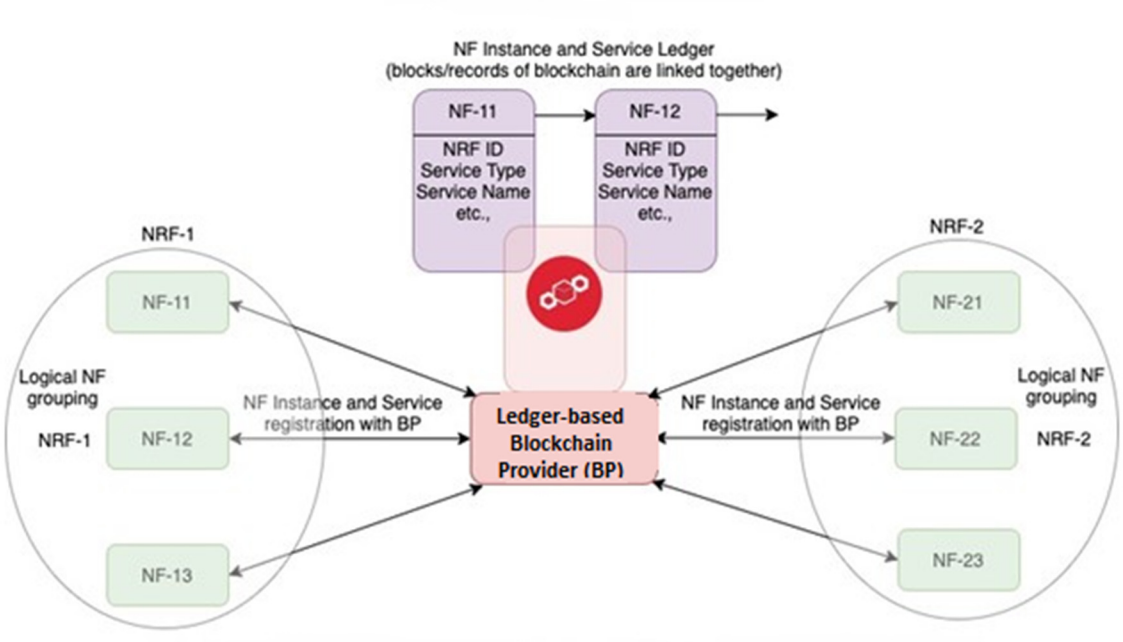

Figure 2: NF Instance and Service Discovery Using Ledger

For simplicity of exposition, the model that is depicted in Figure 2, above, indicates multiple NFs per NRF and multiple NRFs belonging to the same PLMN. However, it is important to note that the same NRF may be part of multiple PLMNs.

As described and illustrated above, according to aspects of the techniques presented herein all of the NF instances in a 5G network deployment are enabled with Ledger (i.e., permissioned blockchain) functionality. In such a permissioned blockchain, the NF instances authenticate with a BP using secure credentials in order to allow only authenticated and trusted NF instances to participate and share the services. Along with authentication, trustworthiness of a NF instance is established using an attestation protocol as described above.

Network instances which are registered with a BP are added as legitimate blockchain entities (i.e., immutable records) in the Ledger. A BP maintains a list of all of the NF instances that are registered in the distributed ledger and helps with, for example, accounting and lawful intercept.

Similarly, whenever a service comes up on a NF instance it will be registered to the blockchain by the respective NF service along with a service type. Note that only authenticated and trusted NF instances may access the permissioned ledger to update the

service. Each block in the Ledger contains the identities of a NF instance and the service that is provided by the NF instance.

Merkle tree hash algorithms (such as the Secure Hash Algorithm 2 (SHA-2) with a 256 bit digest (SHA-256)) may be used for the generation of public keys for enhanced security. Enterprise consensus algorithms such as PoET or PBFT may be used to synchronize databases among all instances in a 5GC.

Use of aspects of the techniques presented herein offers a number of advantages, including the easy, secure and automated addition and removal of services; the faster convergence of a NF instance and service discovery process; a built-in failover mechanism due to the distributed nature of a Ledger and a blockchain; and since the location of a service is not static, dynamic changes to same are automatically learned.

The techniques presented herein may be applied in or to a number of different use cases. For example, aspects of the presented techniques are very much required in large 5G deployments with many NF instances and their services. Further, aspects of the presented techniques may also be adopted in a microservices architecture where services are running on remote nodes especially in a virtual, container, or cloud deployment. Additionally, aspects of the presented techniques are applicable to generic 5G deployments where NF instances and services need to be discovered without having a centralized Service Registry (i.e., where a NRF is acting as Service Registry).

In summary, techniques have been presented herein that employ a private and permissive blockchain-based distributed ledger (which for simplicity of exposition may be referred to herein as a Ledger) to discover NF instances and their services in 5G network deployments. Aspects of the presented techniques decentralize the NF instance and service discovery process without compromising authentication or security. According to aspects of the presented techniques, when a NF instance comes up it may authenticate itself with a blockchain provider to confirm that it is a legitimate NF instance, after which the NF instance and its services may be added to the Ledger. Further, a Ledger is made available to all of the NF instances so that a NF instance may learn about the other NF instances and their services.

Appendix A

Client-Side Discovery

When using client-side discovery, the client is responsible for determining the network locations of the available service instances and then load balancing requests across those instances. The client queries a Service Registry, which is a database of available service instances. The client then uses a load balancing algorithm to select one of the available service instances and then issue a request.

The network location of a service instance is registered with the Service Registry when the instance starts up and it is removed from the Service Registry when the instance terminates. The registration of service instance is typically refreshed periodically using a heartbeat mechanism.

Some video streaming services utilize client-side discovery. Within this context, a REST API can be provided for managing service instance registration and for querying available instances. An inter-process communication (IPC) client may also be utilized to load balance requests across the available service instances.

Server-Side Discovery

Under this model, the client makes a request to a service through a load balancer. The load balancer queries the Service Registry and then routes each request to an available service instance. As with client-side discovery, service instances are registered and deregistered with the Service Registry. Elastic load balancers are one example of a server-side discovery router, such that an elastic load balancer is commonly used to load balance external traffic from the Internet.

Service Registry

A Service Registry is a key part of service discovery. It is a database containing the network locations of service instances. A Service Registry needs to be highly available and it need to be current (i.e., kept up to date). Clients may cache network locations that are obtained from a Service Registry. However, that information eventually becomes stale (i.e., out of date) and clients become unable to discover service instances. Consequently,

a Service Registry consists of a cluster of servers that use a replication protocol to maintain consistency.

Consensus finality property

Under a consensus finality property, once a valid block is appended to a blockchain (at some point in time) that block never was abandoned from the blockchain.

Byzantine Fault Tolerant (BFT)-based Blockchain

A BZT-based blockchain satisfies the consensus finality property and also supports excellent network performance and thousands of NF instances and services (requirements that are significant in 5G network deployments). Additionally, BFT-based blockchain protocols provide excellent performance on throughput.